(1)

NUMERICAL SOLUTION OF THREE-DIMENSIONAL UNSTEADY VISCOUS FLOWS

ONR Grant No: N00014-89-J-1319

Progress Report for the Period

June 1, 1991 - November 30, 1991

Prepared by

Lakshmi N. Sankar
Professor, School of Aerospace Engineering
Georgia Institute of Technology, Atlanta, GA 30332

INTRODUCTION

In many fluid dynamics problems of interest to U. S. Navy, the flows are three-dimensional, unsteady and turbulent. Flow past submarine configurations, flow through marine propellers and turbomachinery are examples of such flows. Numerical procedures for accurate and efficient computations of such flows are presently not possible due to the mixed elliptic-parabolic nature of the governing equations. Indeed, methods for 3-D incompressible flows lag behind 3-D compressible flows by several years. Until accurate and efficient methods for 3-D incompressible, unsteady flows become available, it will not be possible to attempt challenging problems such as first principles based direct simulations of turbulent flow over marine vehicles.

During the past year, under the support of the Office of Naval Research, a research program dealing with numerical solution of 3-D incompressible viscous flows has been underway at Georgia Tech. This report documents progress made under the above project during the period June 1 - November 30, 1991.

OBJECTIVES OF THE PRESENT EFFORT

The long term objective of the present effort is the development of solution techniques for direct numerical simulation of unsteady 3-D incompressible turbulent flows. The kinetic aspects of this problem are governed by a set of parabolic partial differential equations, which may be efficiently integrated by a variety of time marching schemes. The kinematic aspects of this flow such as the relationship between velocity and vorticity, and the relationship between velocity and pressure are governed by elliptic partial differential equations, which can be solved at any instance in time, only by iterative techniques. Direct and/or large eddy simulation of turbulent flows over submarine configurations, turbomachinery, pumps, ducts and other configurations of interest to the U. S. Navy require efficient solution methods for solving the governing equations.

The near term objective of the present research is to investigate and develop efficient time marching schemes for integrating the governing equations, and to evaluate the stability and accuracy of the schemes developed by studying a class of

2-D and 3-D unsteady external flows for which good quality experimental and analytical results are available. The schemes developed as part of the present work should meet the following criteria:

a) They should perform efficiently on the current and future generation vector computers, as well as machines that employ scalable, massively parallel processor architecture.

b) The algorithms should allow fourth or sixth order accuracy in space, even on curvilinear, highly stretched grids. High spatial and temporal accuracy will be important for future direct and large eddy simulation of turbulent flow past general geometries.

c) The algorithms should be cast in a moving, body-fitted, curvilinear coordinate system, allowing a variety of 2-D and 3-D, stationary and moving (e.g. rotating propeller blades) to be studied.

OVERVIEW OF THE SOLUTION PROCEDURE

The governing equations and the solution algorithms tested are cast in a moving curvilinear system. For the sake of convenience, the details of the scheme tested are described for 2-D flows in a Cartesian coordinate system.

The goal of the iterative time-marching algorithm tested is to advance the flow properties (p,u,v) from a known time step 'n' to the next time step 'n+1'. Let 'k' be an iteration counter. Then a quantity such as $u^{n+1,k}$ denotes the variable u at the time level 'n+1' and iteration level 'k'. A good starting guess for the flow variables at time level 'n+1' at the start of the iteration process is the values of these variables at the previous time level. That is,

$$u^{n+1,0} = u^n$$
$$v^{n+1,0} = v^n$$
$$p^{n+1,0} = p^n$$

We also define 'delta quantities' Δu, Δv and Δp such that

$$\Delta u = u^{n+1,k} - u^{n+1,k-1}$$

$$\Delta v = v^{n+1,k} - v^{n+1,k-1}$$

$$\Delta p = p^{n+1,k} - p^{n+1,k-1}$$

Thus, the goal of the iterative process at each time step is to drive these delta quantities Δu , Δv and Δp to zero.

An coupled system of equations for these delta variables may now be written. For example, consider the u- momentum equation (with density assumed to be unity):

$$u_t + (u^2)_x + (uv)_y + \partial p/\partial x = \nu\ (u_{xx} + u_{yy})$$

For the sake of illustration, let us assume that a second order accuracy in time is acceptable. Then, the time derivative $\partial u/\partial t$ will be approximated as

$$\partial u/\partial t = (u^{n+1,k} - u^n)/\Delta t$$

The other terms in the above equation will be evaluated at the 'n+1/2' time level:

$$u^{n+1/2,k} = (u^{n+1,k} + u^n)/2$$

$$u^{n+1/2,k-1} = (u^{n+1,k-1} + u^n)/2$$

The spatial discretizations may be carried out using either a second order accurate central/upwind difference form or a higher order form.

If the quantities such as u^2 , uv and p appearing in the above discretization are linearized about known information u^n and $u^{n+1,k-1}$, then a difference equation linking Δu , Δv and Δp results. Such an equation is given for the u- momentum equation below:

$$\Delta u/\Delta t + [\delta_x(2u^{n+1/2,k-1}\Delta u) + \delta_y(u^{n+1/2,k-1}\Delta v + v^{n+1/2,k-1}\Delta u)] + \delta_x(\Delta p)] /2$$

$$-\nu/2\,(\delta_{xx}+\delta_{yy})\,\Delta u = -[(u^{n+1,k-1} - u^n)/\Delta t + \{\delta_x(u^2) + \delta_y(uv) + \delta_x p$$

$$-\nu(\delta_{xx}u + \delta_{yy}u)\}^{n+1/2,k-1}]$$

Here δ_x , δ_y , δ_{xx} etc. stand for suitable, high order upwind or central approximations to the spatial derivatives.

Note that the right side of the above equation is simply the Crank-Nicholson approximation to the u- momentum equation. If the right side is driven to zero, then the unsteady u- momentum equation will be fully satisfied at the current time level $n+1$.

A similar equation may be written for the v- momentum equation, linking the quantities Δu , Δv and Δp. In the case of continuity equation, one can draw upon the Marker and Cell approach, to link the iterative changes in pressure to changes in velocity, and write

$$\beta\,\Delta p = -(\delta_x u + \delta_y v)^{n+1,k-1}$$

Here β is a free parameter, that may even vary from node to node. It should be noted that the addition of $\beta\,\Delta p$ to the left side of the above equation is not equivalent to a pseudo-compressibility approach. As long as Δp is driven to zero, the discretized form of the continuity equation is exactly satisfied at each time step.

Applying the above discretizations in time and space at all the nodes in the flow field, a system of simultaneous equations results for the quantity Δq equal to (Δu , Δv, Δp). This system may be formally written as:

$$[A]\,\{\Delta q\} = \{R\} \qquad (1)$$

Here, the right hand side is the governing equation, with the temporal and spatial derivative approximated as discussed above. The right side also contains the time derivatives that appear in the governing equation. In traditional iterative

schemes such as the pseudocompressibility scheme, the right side contains only the spatial derivatives. Thus, in these schemes, only the steady state solution is guaranteed. In the present approach, the time accurate solution at each time step is guaranteed, if the right side can be driven to zero.

The matrix A is a sparse, banded matrix whose elements are 3x3 (4x4 in 3-D) matrices, if standard central difference formulas are used to approximate the spatial derivatives. Direct inversion of this matrix requires a huge number of arithmetic operations, despite its sparsity. A common strategy in iterative solutions of elliptic equations is to approximate the matrix A by another, easily inverted matrix B. The closer the matrix A is to B, the faster the iterative convergence of the solution at any time step.

During the reporting period, we tested a B matrix that contains only the diagonal contributions of matrix A. Since the inversion of a diagonal matrix is trivial, the equation system is easily inverted. The solution procedure also exhibits volume parallelism. That is, the flow properties at each and every node in the flow field may be updated in parallel. The algorithm performs efficiently on Cray Y/MP class of machines, but should work equally well on parallel architectures. The price for this simplicity is the large number of iterations needed at each time step (typically 20 to 25, for Reynolds numbers of the order of 1,000,000).

SUMMARY OF SIGNIFICANT RESULTS TO DATE

The algorithm described above has been implemented both in a two-dimensional Navier-Stokes solver and in a 3-D Navier-Stokes solver.

Two-Dimensional Results: The iterative algorithm described above was tested by computing unsteady laminar viscous flow past a sinusoidally pitching NACA 0012 airfoil, at a Reynolds number of 5,000. This case has been previously studied by Mehta at NASA Ames Research Center using a velocity-vorticity formulation. Figure 1 shows the body-fitted grid around the airfoil used in this study. Figure 2 shows the variations in lift, drag and pitching moment as a function of angle of attack as the airfoil pitches up to 20 degrees and returns to zero degree. A massive, highly unsteady, separated flow over the airfoil occurs during this maneuver. Thus, this case provides a good test of the present algorithm's ability to maintain time

accuracy. Figure 3 shows the streamlines, velocity vectors over the airfoil, vorticity contours and surface pressure distribution at several instances in time. For the sake of comparison, the surface pressure distributions of Mehta, computed using a vorticity-stream function formulation is shown. In general, excellent agreement was found between the computed results and Mehta's solution.

Three-Dimensional Results: A three-dimensional, incompressible Navier-Stokes method, capable of predicting massively separated flow over bluff configurations such as an ellipsoid of revolution at an angle of attack has also been developed. Like the 2-D solver, this method allows the body to move in a very general fashion and undergo pitching, plunging and yawing motions. The solution procedure is third order accurate in space, and uses an upwind scheme. Second order accuracy in time is possible.

This solver was tested by computing the flow past an ellipsoid of revolution at 10 degree angle of attack, at a Reynolds number of 5,000. Figure 4 shows the body-fitted grid used in the study. Figure 5 shows the particle traces over the body surface, and the velocity vector field in the immediate vicinity of the body. There is a limited amount of experimental data available for this particular configuration, at a high turbulent Reynolds number. Figure 6 shows the surface pressure distribution on the windward and leeward sides of the symmetry plane, along with the experimental data. Good agreement is evident everywhere except in the last 10% of the body, where the present laminar simulation predicts flow separation, and a flattening out of the pressure distribution.

Acceleration of 2-D Unsteady Flow by Multigrid Techniques: The multigrid technique was developed during the 1970s as a technique for accelerating iterative numerical solution of elliptic partial differential equations. Since that time, this technique has been applied to a variety of fluid flow problems including steady transonic potential flow, and 2-D and 3-D inviscid rotational flows.

During the reporting period, it was investigated whether some of the iterations for u, v and p at a time step 'n+1' can be done inexpensively on a coarse grid (where fewer grid points exist), without sacrificing the fine grid accuracy of the numerical solution. This is equivalent to computing a first estimate for the quantity q^{n+1} appearing on a coarse grid. The following procedure was developed to advance

the flow properties q at time level 'n+1', at iteration level 'k', $q^{n+1,k}$ to the next iteration, $q^{n+1,k+1}$.

i) Compute the residual {R} appearing on the right side of equation (1) on the fine grid using $q^{n+1,k}$

ii) Transfer the residual from the fine grid to a coarse grid using the injection operation, $I_h^{2h}R$. A typical injection operation that is easy to implement is given at any node (i,j) by

$$I_h^{2h}R_{ij} = R_{i,j} + (R_{i+1,j}+R_{i-1,j}+R_{i,j+1}+R_{i,j-1})/2 + (R_{i+1,j-1} + R_{i-1,j+1} + R_{i-1,j-1} + R_{i-1,j+1})/4$$

iii) Compute the quantity Δq at every point on the coarse grid by solving the system of equations:

$$[B]\{\Delta q\} = \{I_h^{2h}R\}$$

As mentioned earlier, in our present implementation, the matrix [B] is just the diagonal portion of the matrix [A].

iv) Interpolate the Δq values computed in step (iii) back onto the fine grid.

v) Compute the updated values of the flow properties $q^{n+1,k+1}$ as $q^{n+1,k} + \Delta q$

Repeat step (i) - (v) till Δq is driven to zero. The resulting coverged solution q^{n+1} forms an excellent starting guess for subsequent fine grid iterations.

At first glance, no CPU time appears to have been saved, because the residuals in step (i) need to be computed at all fine grid node points. Indeed, there are no significant CPU time savings per iteration in the multi-grid method just outlined. But the savings come in the form of larger time steps that may be used without instability, improved starting guesses for q on the fine grid, and fewer iterations on the fine grid to drive {R} to zero.

The multigrid process described above has been implemented in the 2-D unsteady viscous flow analysis. For steady flow applications, where an asymptotically

steady state solution is reached after several time steps, the multigrid process reduced the overall CPU time by over 30%, compared to a single grid iteration procedure.

ANTICIPATED RESULTS FOR THE NEXT REPORTING PERIOD

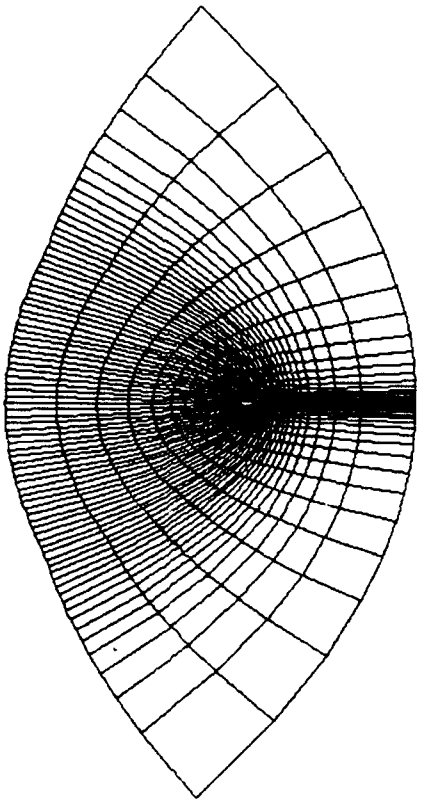
By the conclusion of the next reporting period (May 31, 1992), we plan to have the following work completed.

a) In all the work done, the matrix B (which is an approximation to the matrix A) is a simple diagonal-matrix. While use of such a simple diagonal matrix simplifies the inversion, and makes the flow solver 100% vectorizable, it leads to slow convergence of the pressure and velocity fields at every time step. We will be investigating alternate B matrices, for example, a B matrix that is constructed by transferring the system of equations (1) to a coarser grid using classical multi-grid techniques. Another choice for the B matrix is an LU approximation to the A matrix. Inversion of LU matrices can be done on vector machines efficiently, if the equations are properly ordered, so that the elimination is done along diagonal lines or planes.

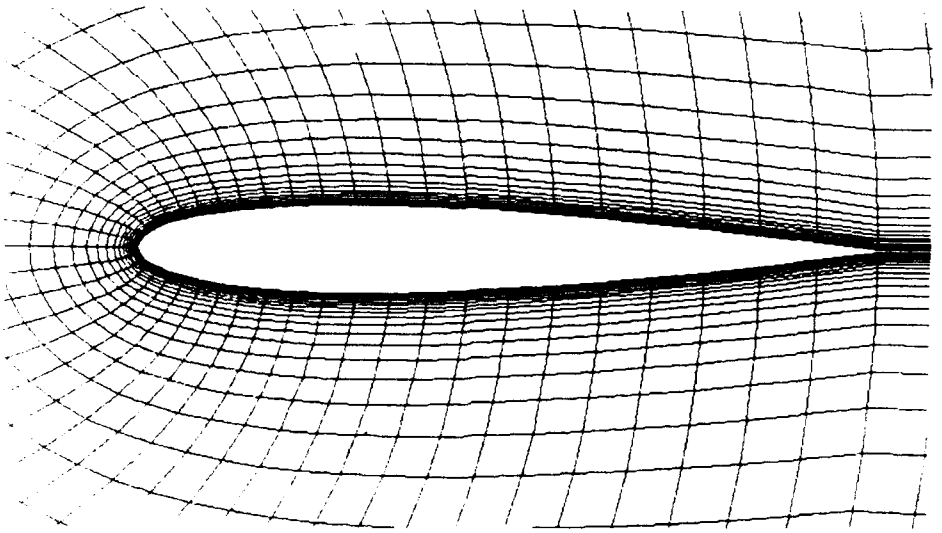
b) The equation set (1) may be inverted using classical conjugate gradient schemes. We plan to investigate a class of conjugate gradient methods, known as the Generalized Minimum Residual method (GMRES) for the efficient inversion of the solution scheme. We have experience using the GMRES algorithm in another (compressible) flow solver, and found the flow solver to yield a factor of 4 speed up compared to classical ADI methods, when applied to unsteady viscous flows.

CONCLUDING REMARKS

It is anticipated that the present efforts will lead to a fairly general, and efficient ways of solving the 3-D incompressible Navier-Stokes equations. The resulting methods will provide a very good starting point for more ambitious efforts such as direct numerical simulation of turbulent flow over 3-D submarine configurations.

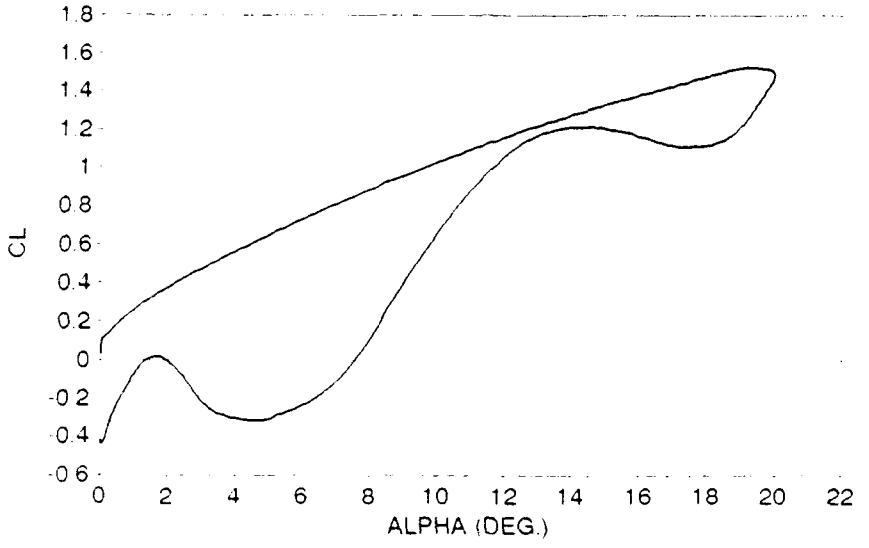


(a)

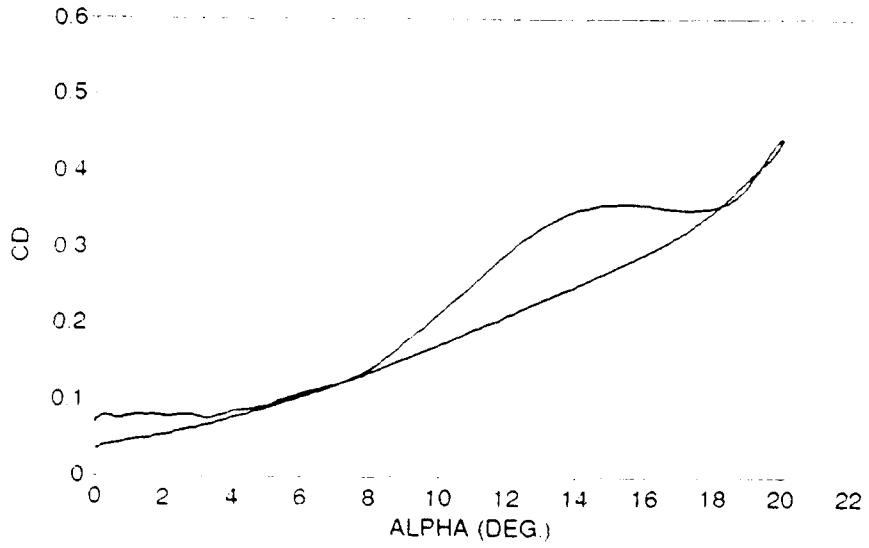


(b)

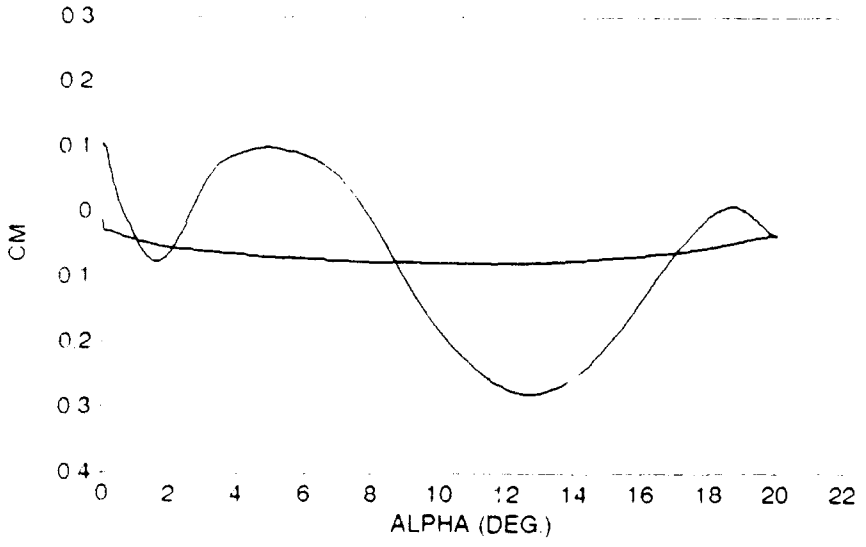
Figure 1. Body-Fitted Grid Around a NACA 0012 airfoil



(a)

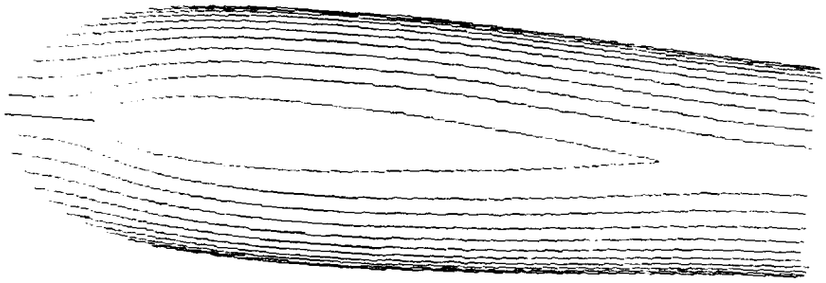


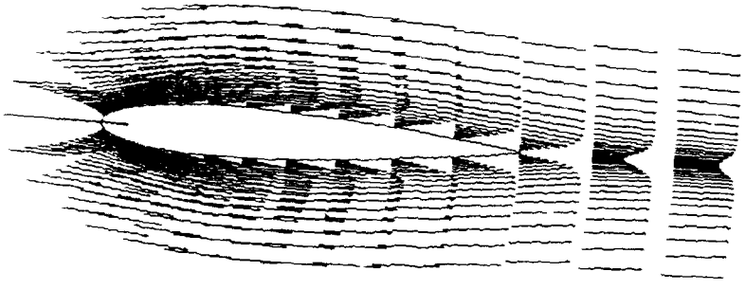
(b)

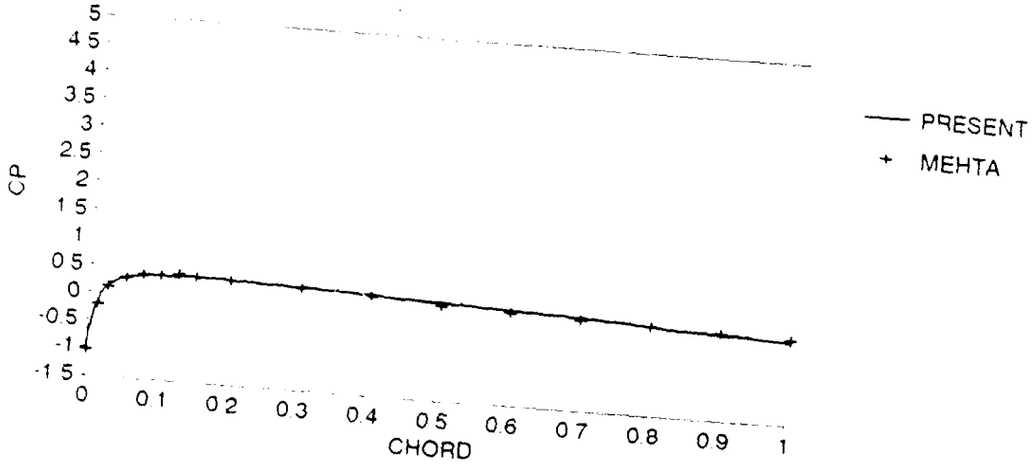


(c)

Figure 2. Dynamic Stall Hysteresis Loops for a NACA 0012 Airfoil

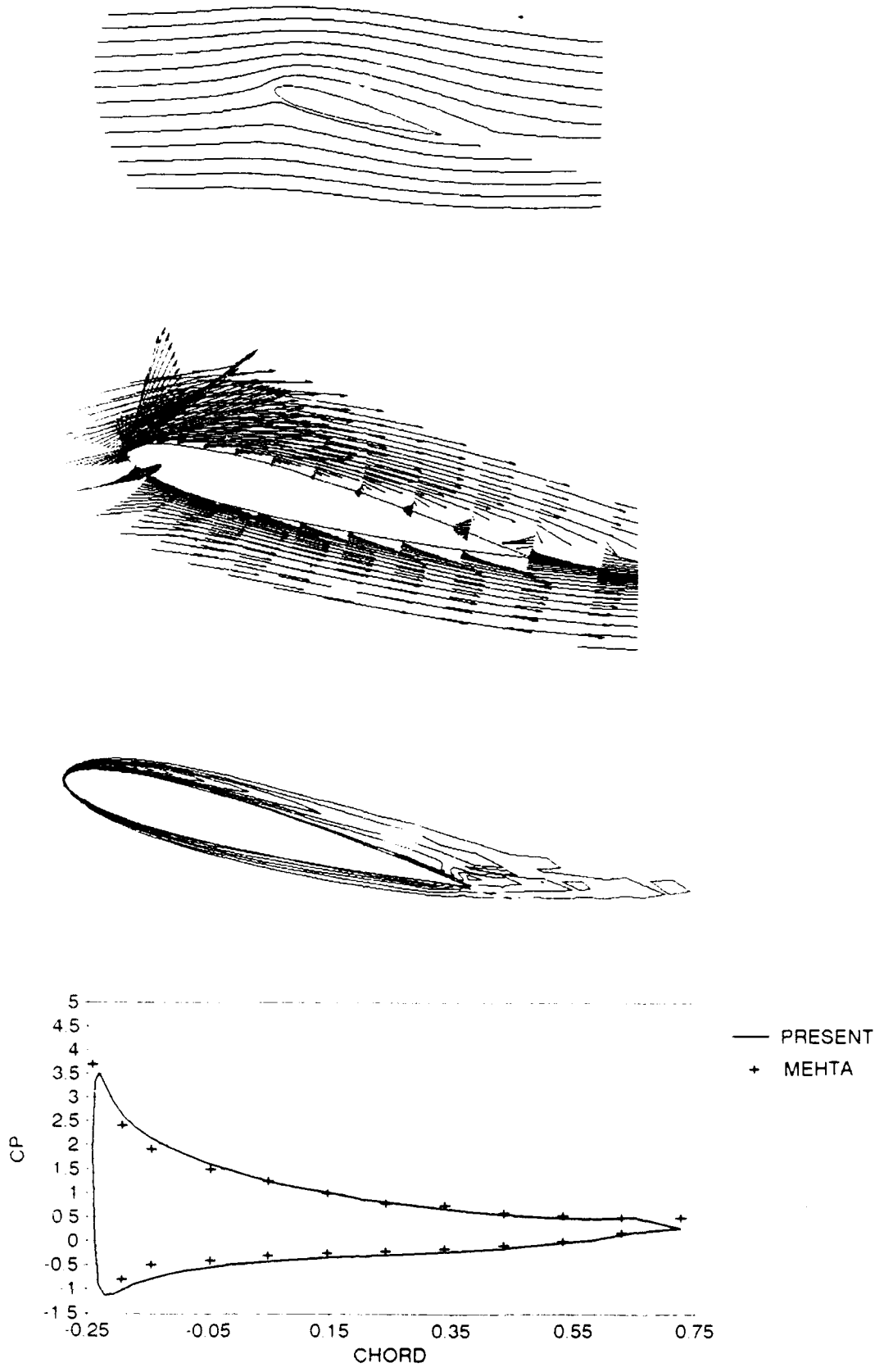






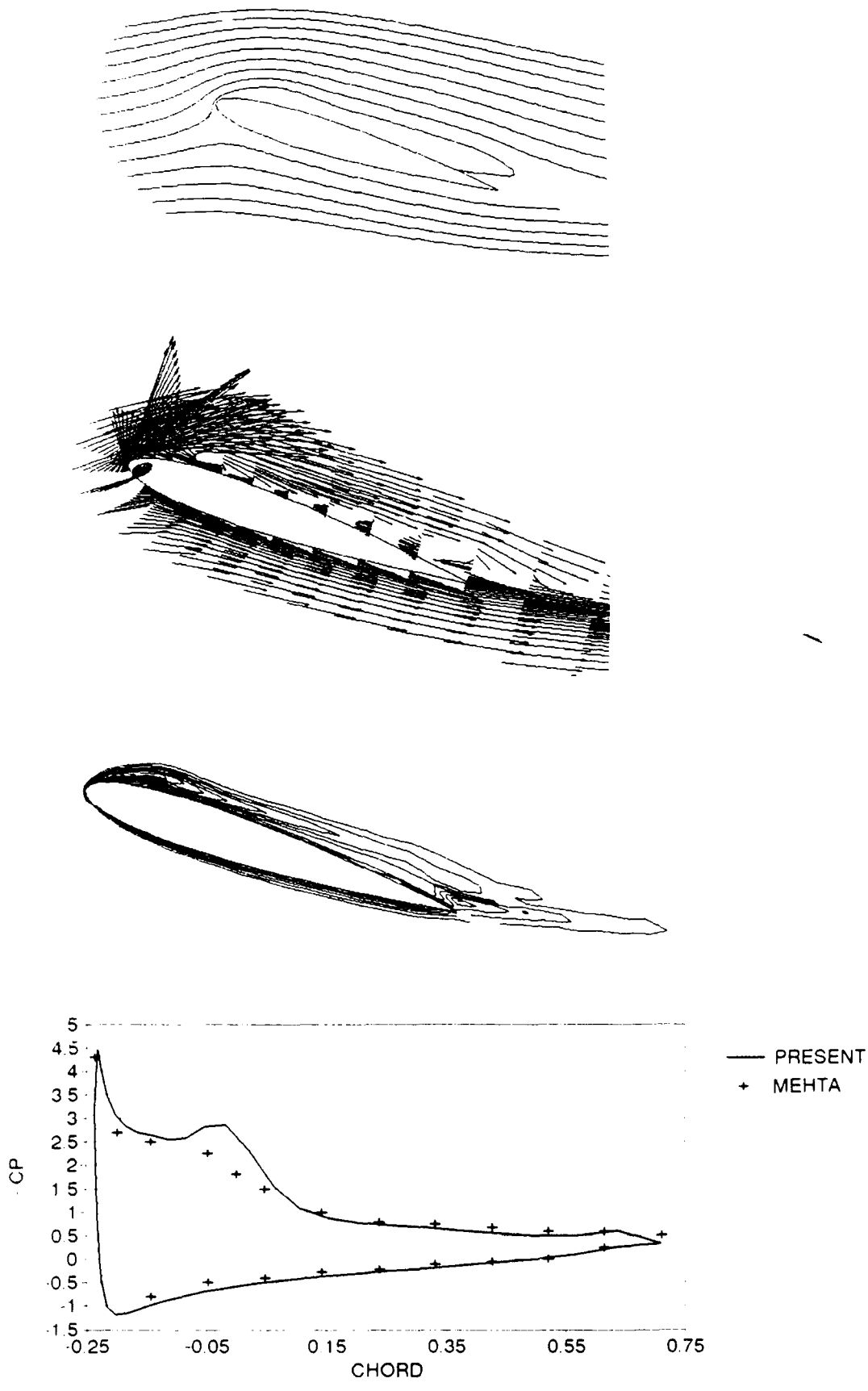
(a) $\alpha = 0$

Figure 3. Streamlines, Vorticity , Velocity Field and Surface Pressures over an Oscillating Airfoil at Selected Angles of Attack



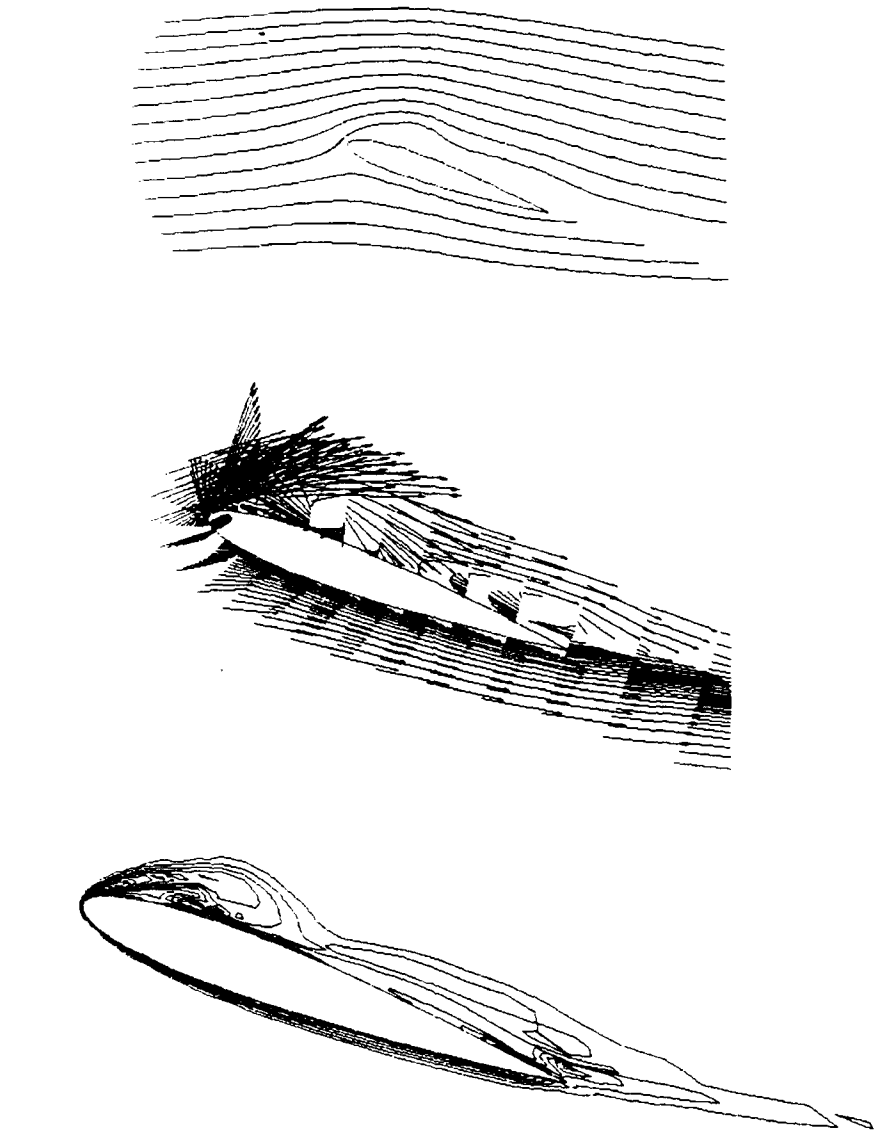
(b) $\alpha = 14.6$

Figure 3. Continued.



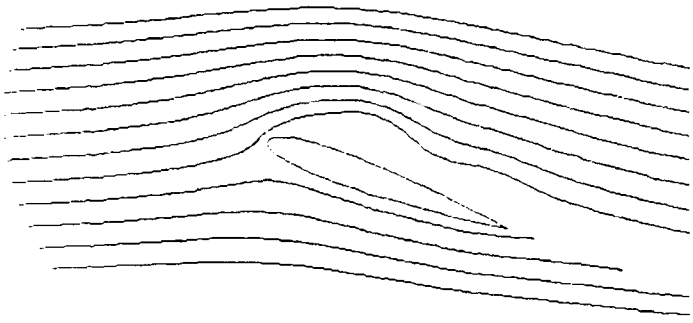
(c) $\alpha = 18.8$

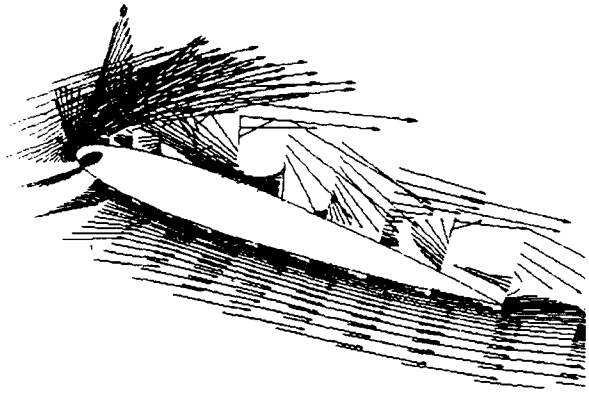
Figure 3. Continued.

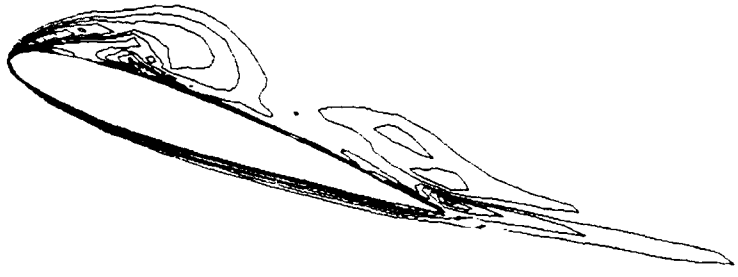


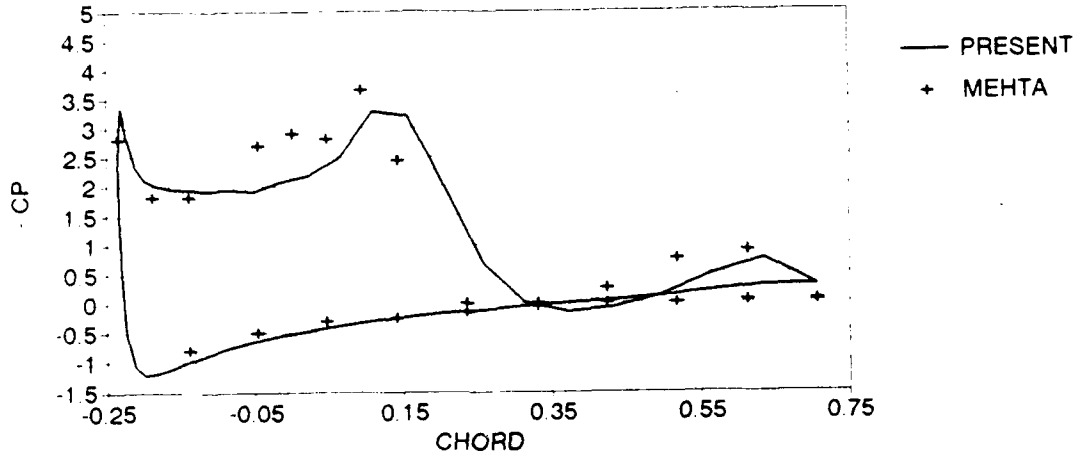
(d) $\alpha = 20$

Figure 3. Continued.



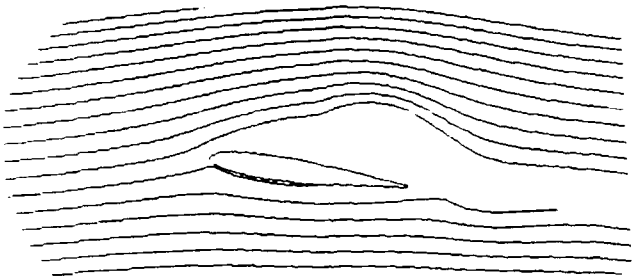


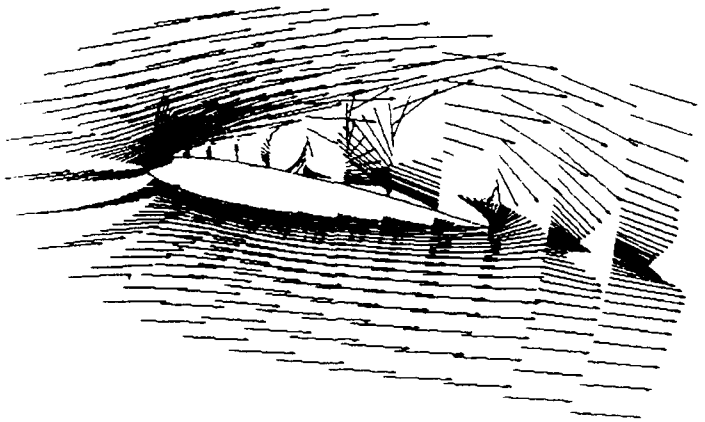


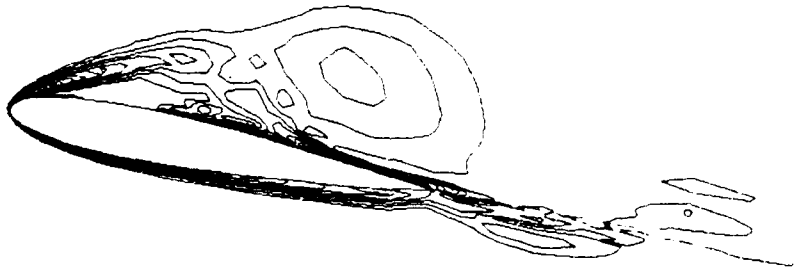


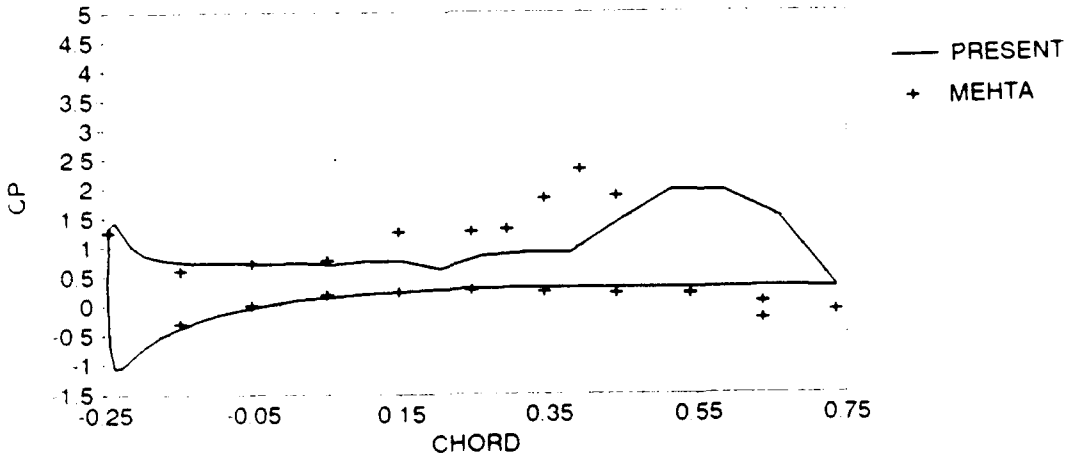
(e) $\alpha = 19.8$

Figure 3. Continued.



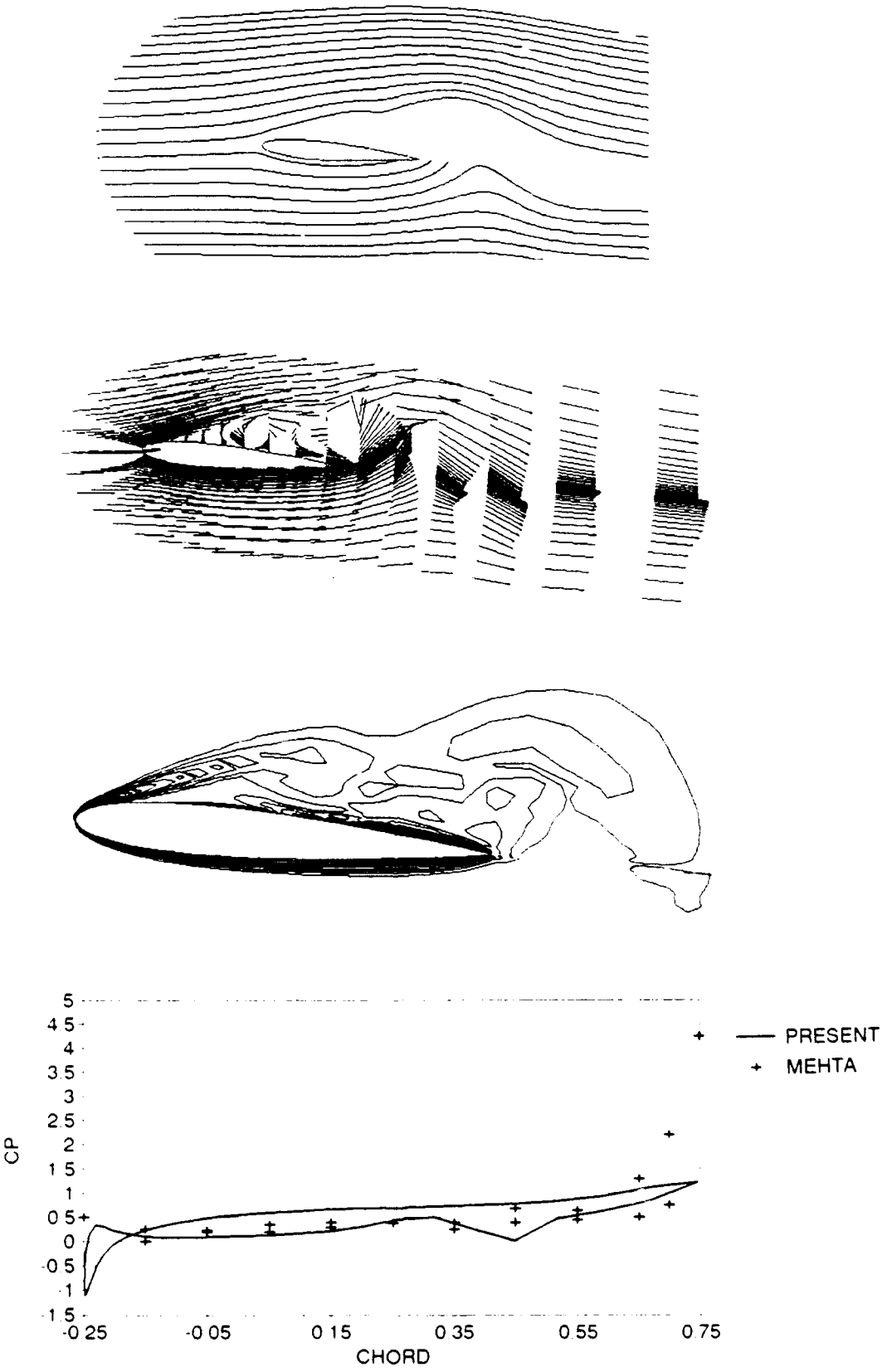






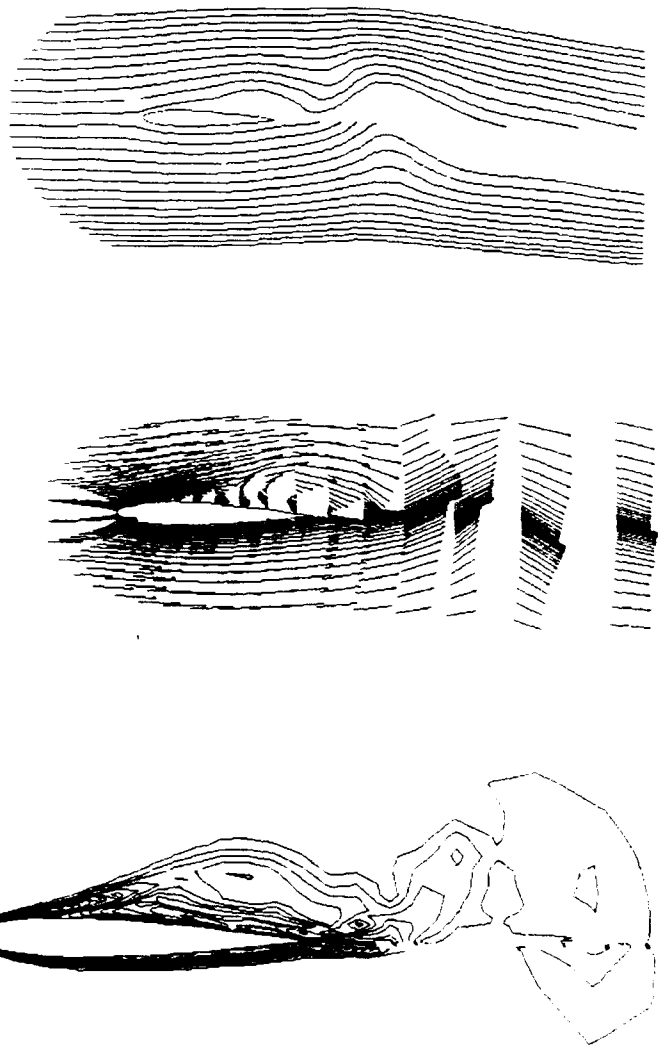
(f) $\alpha = 11$

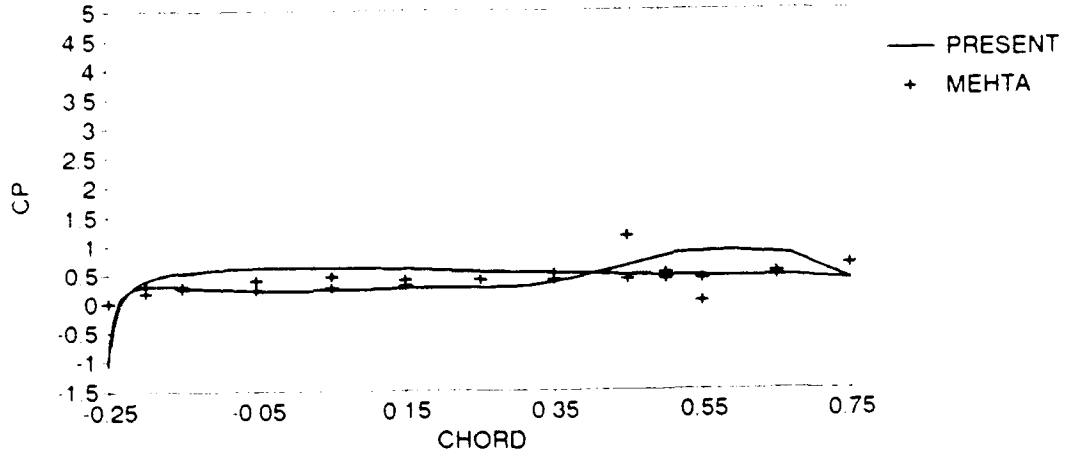
Figure 3. Continued.



(g) $\alpha = 5$

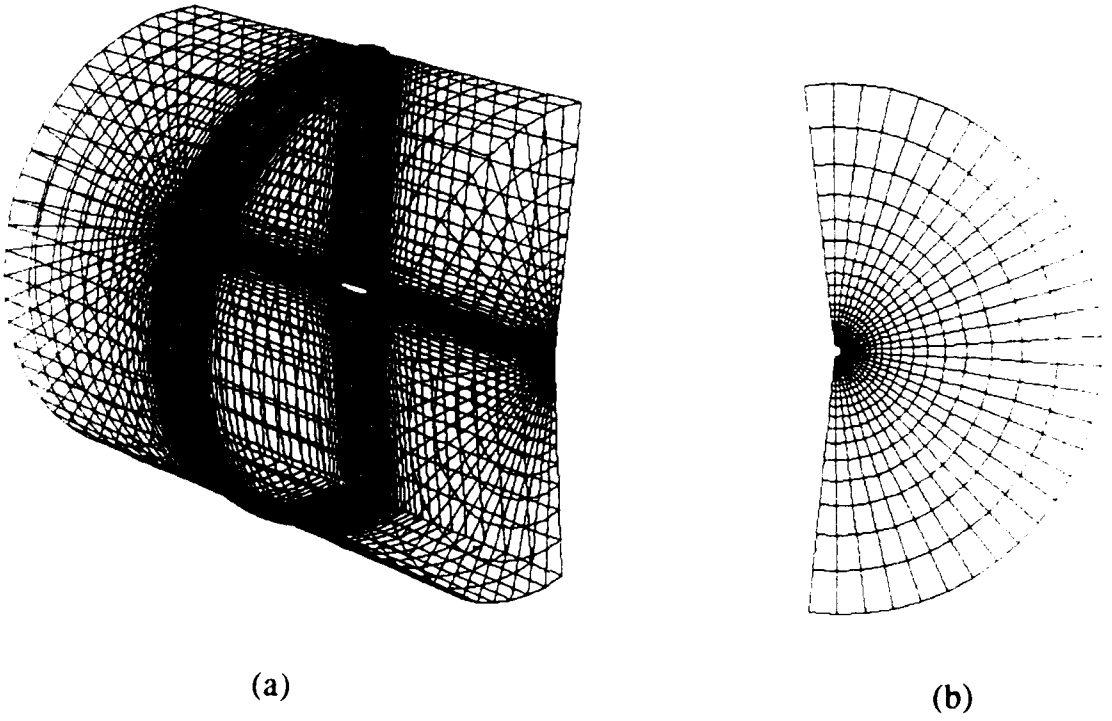
Figure 3. Continued.



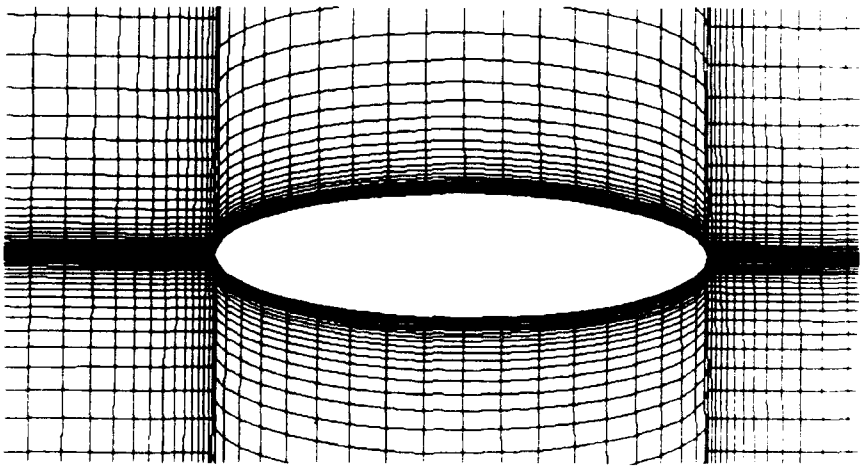


(h) $\alpha = 1$

Figure 3. Continued.

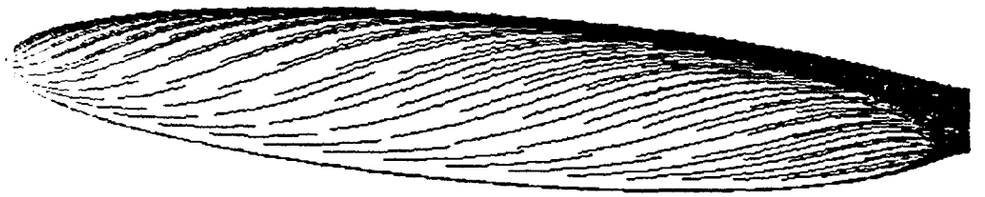


(a) (b)

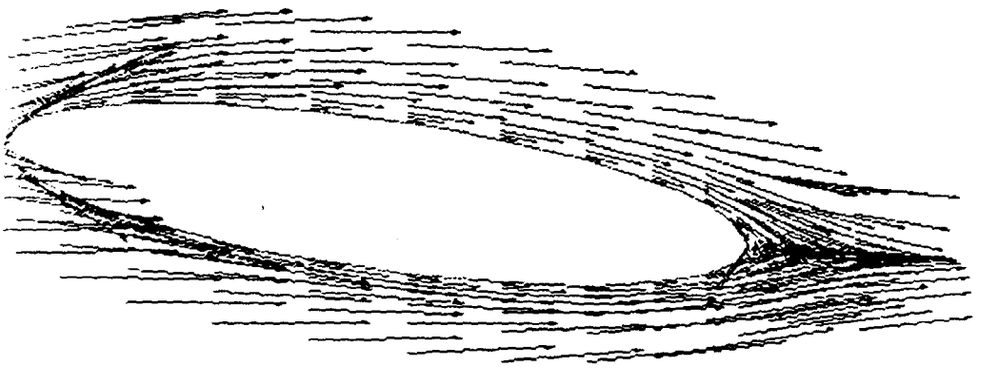


(c)

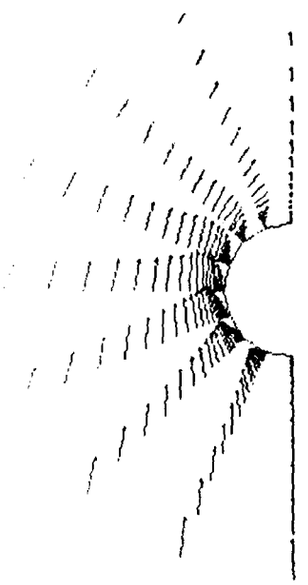
Figure 4. Body-Fitted Grid Around an Ellipsoid of Revolution



(a)

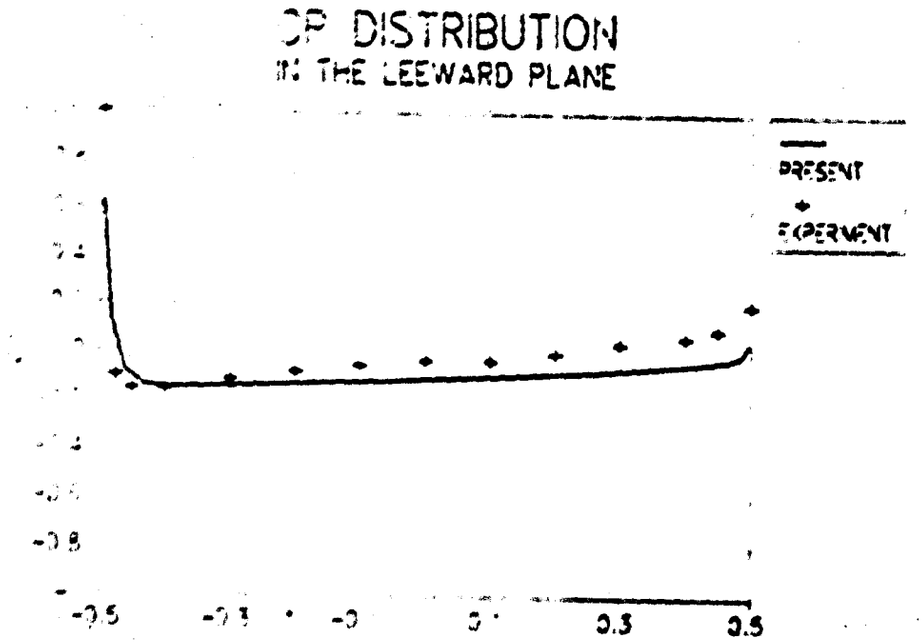


(b)

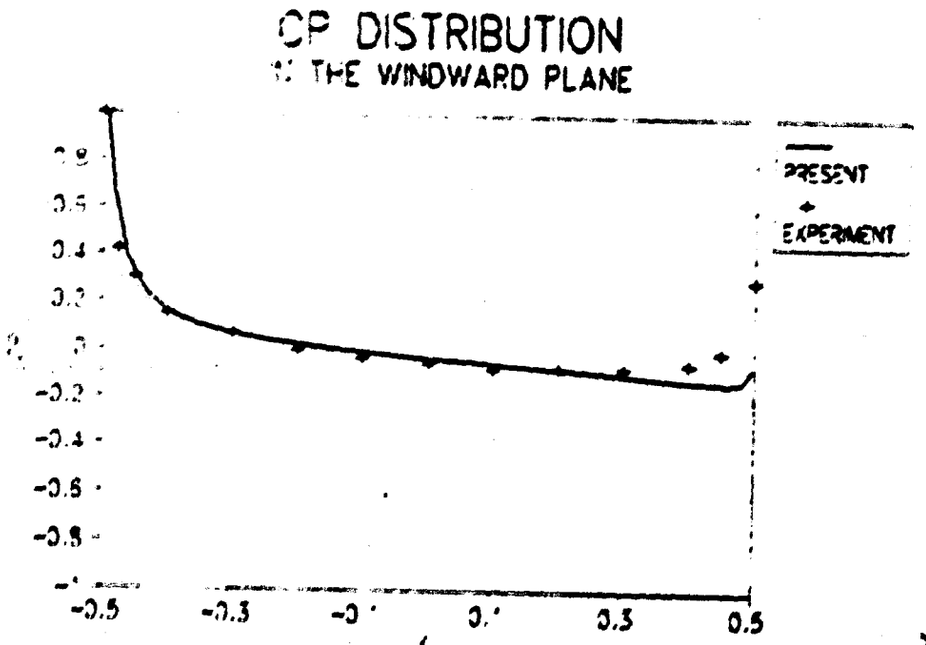


(c)

Figure 5. Particle Traces, Streamwise and Crossflow Velocity Vectors over the Ellipsoid of Revolution, 10 Degree Angle of Attack, Reynolds Number = 5,000



(a)



(b)

Figure 6. **Comparsion of Computed Surface Pressures with Experiments by Meier et. al.**